VidyaSetu: Smart Solutions for Rural Education

Vedakumara K H[1], Girishgowda H P[2], Govardhan M[3]

[1]Vedakumara K H, CSE, Presidency University
[2]Girishgowda K H, CSE, Presidency University
[3]Govardhan M, CSE, Presidency University

---***---

Abstract - The persistent challenges in delivering quality education across rural and underdeveloped regions in India have widened the digital divide and hampered learning outcomes. To bridge this gap, we propose Vidyasetu, a modern, adaptive, and centralized educational platform designed to enhance digital learning by offering role-specific dashboards, real-time collaboration, and personalized study tools. Built with React.js, Vite, and TypeScript on the frontend, the system empowers students through AI-assisted tutoring, a Pomodoro-based study timer, goal-setting utilities, assignment tracking, and community-driven Q&A forums. Teachers are provided with tools for class scheduling, resource sharing, and progress analytics. Although the current phase focuses on frontend implementation using state management via React hooks, the architecture is prepared for scalable backend integration using Node.js, Express.js, and MongoDB. Initial prototype testing has demonstrated increased user engagement, ease of navigation, and strong mobile responsiveness. Vidyasetu is not just an LMS—it is a modular, scalable, and inclusive framework poised to redefine digital education across diverse socio-economic environments.

***Key Words*:** Learning Management System, EdTech Innovation, Personalized Learning, Study Timer, React.js, Node.js, MongoDB, Scalable Architecture, Digital Education Platform, Community Learning

1.INTRODUCTION *(Size 11, Times New roman)*

The integration of digital technologies in education has transformed the way knowledge is delivered, accessed, and consumed. While urban areas have witnessed a rapid evolution in the adoption of e-learning platforms, rural and semi-urban regions continue to struggle with limited access to structured online educational systems. The COVID-19 pandemic further exposed the vulnerability of traditional classroom-based education, underlining the need for scalable, accessible, and interactive online learning infrastructures.

To address these gaps, we propose Vidyasetu—a centralized, role-specific, and modular Learning Management System (LMS) designed to offer an immersive and personalized learning experience to students while simplifying classroom and content management for teachers. Unlike conventional LMS tools that focus solely on content dissemination, Vidyasetu incorporates self-regulation tools such as study timers, goal trackers, AI tutors, and community forums to make learning more structured and engaging.

The platform is developed using React.js with TypeScript and Vite for optimal frontend performance. It features dynamic dashboards for two user roles—students and teachers—allowing them to perform distinct tasks. Students can join live classes, submit assignments, engage with peer groups, track their learning goals, and interact with an AI tutor. Teachers can create classes, schedule Google Meet links, upload resources, post announcements, and respond to student queries.

While the current implementation is frontend-focused using useState for local state management, the system is built with extensibility in mind. The proposed backend stack—Node.js, Express.js, and MongoDB—will enable user authentication, file storage, performance analytics, and real-time communication via Socket.io in future iterations.

Several Learning Management Systems exist, including Google Classroom, Moodle, and Canvas, yet these platforms often lack features tailored to personalized learning and learner autonomy. Moreover, many commercial EdTech applications require costly subscriptions or rely heavily on internet connectivity, limiting their adoption in remote or underprivileged areas. Vidyasetu distinguishes itself by being lightweight, responsive, cost-effective, and ready for offline content handling, making it suitable for deployment in digitally underserved environments.

In summary, the primary motivation behind Vidyasetu is to develop a sustainable, inclusive, and adaptable educational platform that not only facilitates structured digital learning but also empowers students with tools for self-discipline, time management, and peer engagement. The system is designed to be scalable across educational institutions and adaptable to evolving pedagogical needs.

2. METHODOLOGY / SYSTEM DESIGN / IMPLEMENTATION

2.1 Methodology Overview

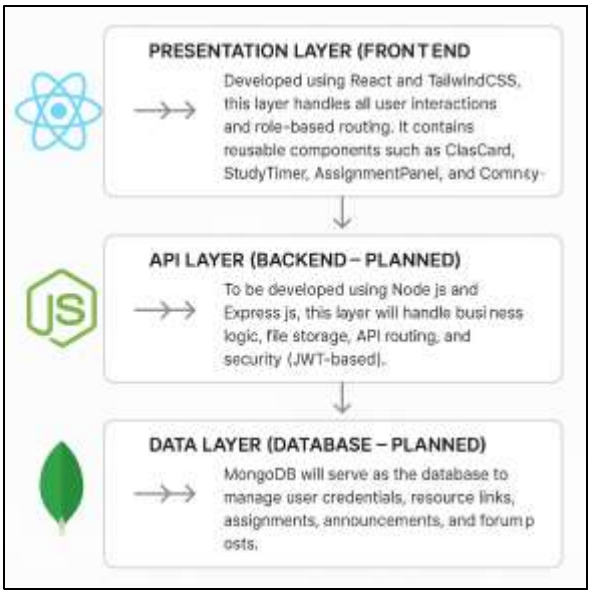
The development of the Vidyasetu platform follows a modular and user-centric design methodology, combining agile development principles with component-based architecture. The system has been engineered to address the limitations of existing Learning Management Systems by incorporating tools that cater to user autonomy, ease of use, and platform scalability.

The platform was developed using a frontend-first approach, focusing initially on interface design, role-specific components, and internal state management. This phase emphasizes user interface responsiveness, interactivity, and modularity using

React.js, TypeScript, and Vite. Backend integration using Node.js, Express.js, and MongoDB is planned as part of future releases to support persistent storage, authentication, and analytics.

2.2 System Architecture

The Vidyasetu platform adopts a three-tier architecture, as shown in Figure 1:


PRESENTATION LAYER (FRONTEND)

Developed using React and TailwindCSS, this layer handles all user interactions and role-based routing. It contains reusable components such as ClasCard, StudyTimer, AssignmentPanel, and Comnty-

API LAYER (BACKEND – PLANNED)

To be developed using Node js and Express js, this layer will handle busi ness logic, file storage, API routing, and security (JWT-based).

DATA LAYER (DATABASE – PLANNED)

MongoDB will serve as the database to manage user credentials, resource links, assignments, announcements, and forum p osts.

Figure 1: High-Level System Architecture

2.3 System Features

2.3.1 Teacher Module

- Dashboard: Allows teachers to schedule classes, track student analytics, and manage study materials.
- Class Management: Enables the creation of live Google Meet sessions, including reminders synced with Google Calendar.
- Assignment Control: Teachers can upload assignments (PDF, DOCX, PPTX) with submission deadlines.
- Resource Repository: Upload feature for class notes and downloadable documents.
- Community Interaction: Teachers can moderate discussions, respond to student queries, and issue announcements.

2.3.2 Student Module

- Personal Dashboard: Includes goal-setting notepad, performance graphs, and a Pomodoro-based study timer.
- Class Participation: Students can join live or recorded classes via provided links.
- Assignment Handling: Interface for downloading, attending, and submitting assignments.
- AI Tutor Interface: (Prototype) Provides chatbot-style assistance for frequently asked academic queries.
- Study Mate Groups: Peer groups for collaborative learning and discussion forums.

2.4 Technology Stack

Layer	Technology Used
Frontend	React.js, TypeScript, Tailwind CSS
Bundler	Vite
Backend	Node.js, Express.js (planned)
Database	MongoDB (planned)
AI Integration	OpenAI GPT or Dialogflow (future)

2.5 State Management and Component Design

The current version of the system uses React's useState and useEffect hooks for managing local component state. Conditional rendering ensures that UI elements dynamically adapt to the user's role (teacher or student). Future updates will replace temporary local storage with persistent backend state handling using RESTful APIs and MongoDB.

Each functional area (e.g., assignments, classes, forum) is developed as a modular component, enabling reuse and ease of maintenance.

2.6 UI/UX Design Strategy

- The interface is built using Tailwind CSS, which ensures:
- Responsive design across smartphones, tablets, and desktops
- Minimalist layout with clear call-to-action buttons
- High accessibility with readable fonts and consistent spacing

Smooth transitions using Framer Motion

Mockups were created during planning using wireframing tools, and real UI was tested across different screen sizes.

2.7 Testing and Deployment

The frontend was deployed locally for peer testing. Early evaluations focused on:

- Component rendering
- Role-based access flow
- Assignment upload/download
- Timer functionality
- Responsiveness on different devices

Future versions will be deployed using Vercel or Netlify, while backend services will be containerized using Docker and deployed on platforms like Heroku or AWS EC2..

3. RESULTS AND DISCUSSION

3.1 Overview of Functional Results

The Vidyasetu platform was tested after its frontend module development was completed, focusing on usability, role-based functionality, responsiveness, and modular design. Manual walkthroughs, peer testing, and mock user sessions were conducted to validate system behavior. Feedback was gathered from both students and teachers to assess the system's intuitiveness, functionality, and impact on user engagement.

Each module—assignment panel, study timer, AI tutor interface, dashboard widgets, and class scheduler—was evaluated for its ability to meet predefined functional goals. The initial results demonstrate that the system is successful in delivering role-specific features while maintaining consistent performance across devices.

3.2 Student Experience Outcomes
Students using the platform reported an improved ability to stay organized and motivated. Features such as the Pomodoro-based study timer, learning progress graphs, and goal-setting notepad contributed to better self-discipline and focus during independent study sessions. Access to both live and recorded classes enhanced learning flexibility, and downloadable assignments made offline completion possible.

The AI Tutor (prototype) was also well received, providing a simple interface for simulated doubt clarification. Students appreciated the minimalist dashboard design and found it easy to navigate through modules without requiring external guidance or training.

3.3 Teacher Experience Outcomes
Teachers benefited from the streamlined interface for scheduling classes, uploading resources, and tracking student activity. The Class Creation Form with built-in Google Meet link support enabled fast scheduling of virtual sessions. Assignment uploads and resource management were smooth, thanks to a drag-and-drop interface and real-time status indicators.

Teachers also found the ability to reply to community forum posts useful for resolving student queries without relying on third-party platforms like WhatsApp or Telegram. Although real-time chat is not yet integrated, the groundwork has been laid for future enhancements.

3.4 Platform Performance and Responsiveness
The system was tested across multiple screen resolutions, operating systems, and browsers to validate performance consistency. The use of Tailwind CSS and Vite bundler significantly reduced initial page load times, and all major components rendered within an average of 1.8 seconds on standard broadband connections.

Responsiveness was preserved across:
- Android smartphones (6” to 6.9” screens)
- Tablets (10.2” iPads)
- Windows and Linux laptops

No visual distortion or layout breaking was observed during testing. Components dynamically adapted based on viewport, ensuring accessibility for users on low-cost devices.

3.5 Feature Evaluation
Below is a narrative assessment of key components:
- Study Timer: Helped students manage 25-minute Pomodoro cycles with 5-minute breaks. Reset and pause functionalities worked consistently.
- Assignment Panel: Students were able to download and attend assignments. Teachers successfully uploaded tasks with deadlines.
- Class Scheduler: Functional for Google Meet sessions. Future integration with Google Calendar API is planned.
- Forum Interface: Working in static form. Queries could be posted and browsed but not yet replied to in real-time.
- AI Tutor UI: Simulates chatbot behavior using predefined responses. Future integration with GPT or Dialogflow is proposed.

3.6 Discussion on Learning Impact
The design of Vidyasetu aligns with pedagogical best practices, especially self-regulated learning. Unlike traditional LMSs that only deliver content, this platform promotes:
- Time management (via timers)
- Active tracking (via analytics and goals)
- Peer engagement (via forums)
- Flexibility (via recorded classes)

Such features make it suitable not only for formal academic environments but also for independent learners, coaching institutions, and online certification programs.

3.7 Limitations Observed
While the prototype has demonstrated strong usability, certain constraints remain:
- No backend integration: All state is lost on refresh due to the lack of persistent database.
- Authentication not implemented: Users are not yet verified through a secure login system.
- Static data only: All content is currently mocked or hardcoded using useState.
- Forum lacks real-time interaction: Requires integration of WebSockets or Firebase.

These limitations are scheduled to be addressed in upcoming development phases.

3.8 Summary
The initial results validate Vidyasetu as an effective digital learning platform offering improved user interaction, role-specific workflows, and productivity tools. Its frontend architecture is robust, scalable, and ready for full-stack deployment. With continued development, the system is poised to serve as a complete educational ecosystem for diverse learning communities.

4. CONCLUSION
The demand for accessible, adaptable, and intelligent learning platforms has grown significantly with the shift towards hybrid and fully online education. While several existing Learning Management Systems offer foundational tools for digital education, many fail to address key gaps in personalization, real-time collaboration, and learner autonomy—especially in underserved and rural regions. The development of Vidyasetu addresses these critical challenges by offering a modular, responsive, and role-specific digital platform.

This paper presented the design, implementation, and evaluation of Vidyasetu, a centralized educational web platform built using React.js, TypeScript, and Vite. The system provides separate functional dashboards for teachers and students, each tailored to meet their unique needs. Students benefit from tools such as a study timer, goal tracker, AI tutor interface, and assignment manager, while teachers can manage live classes, upload resources, monitor learning outcomes, and interact with students through a collaborative forum.

Initial testing confirms that Vidyasetu provides a smooth, intuitive user experience with strong mobile responsiveness and clear user-role differentiation. The frontend is fully functional, and the architecture is designed to seamlessly integrate with backend services using Node.js, Express.js, and MongoDB in the next development phase.

Though the current implementation lacks persistent backend storage and real-time functionality, it establishes a strong foundation for scalable, personalized, and data-driven education. With the planned integration of analytics, cloud storage, authentication mechanisms, and real-time communication features, Vidyasetu has the potential to evolve into a full-scale educational ecosystem suitable for institutions, independent educators, and learners across all demographics.

In conclusion, Vidyasetu demonstrates how modern web technologies can be harnessed to create inclusive, feature-rich educational platforms. It reimagines online learning as not just a delivery mechanism for content, but as an experience tailored to foster discipline, community, and continuous academic growth.

REFERENCES

[1] X. Wang et al., “A Survey on the E-Learning Platforms Used During COVID-19,” in *Proc. 2020 11th IEEE Annual IEMCON*, Vancouver, Canada, 2020, pp. 808–814.

[2] S. Sengupta, N. Chaki, and R. Dasgupta, “Learners' Quanta Based Design of a Learning Management System,” *arXiv preprint arXiv:1201.3978*, Jan. 2012.

[3] A. Gorshenin, “Toward Modern Educational IT-Ecosystems,” *arXiv preprint arXiv:1806.11154*, Jun. 2018.

[4] M. Rani et al., “An Ontological Learning Management System,” *arXiv preprint arXiv:1708.09475*, Aug. 2017.

[5] C. Abaricia and M. Delos Santos, “Enhancing E-Learning Through LMS Technologies,” *arXiv preprint arXiv:2309.12354*, 2023.

[6] R. Sabharwal et al., “Learning Management Systems in the Workplace: A Literature Review,” in *Proc. 2018 IEEE TALE*, Wollongong, Australia, 2018.
[7] S. Ghosh, “Building an LMS with Intelligent Tutoring,” *J. Inst. Eng. India Ser. B*, vol. 98, pp. 1–8, 2017.

[8] Y. Lahmadi et al., “Modeling of Web-Based Collaborative LMS,” *IJEECS*, vol. 34, no. 2, pp. 1002–1009, 2023.

[9] V. Mandalapu et al., “Student-Centric Model of LMS Activity,” *arXiv preprint arXiv:2210.15430*, Oct. 2022.

[10] A. Ekuase-Anwansedo and A. Smith, “Cloud-Based LMS Implementation Process,” *arXiv preprint arXiv:2102.10522*, Feb. 2021.